Crear Autoridad Certificadora

Luis Carlos Leaño Martin - 744732
Ing. Ciberseguridad

ITESO, Universidad Jesuita de Guadalajara

Primero empezamos descargando estos archivos de GitHub de el siguiente link que son root-ca y sub-ca los configuramos con nuestro dominio, la información del país y le ponemos que no a la contraseña ya que por ser por primera ves y para hacerlo mas sencillo lo haremos sin contraseña.

Plantilla de Root-ca y Sub-ca:
-> https://github.com/ivanr/bulletproof-tls/tree/master/private-ca

```
[default]
name                    = root-ca
```

```
domain_suffix           = leanocompany.com
aia_url                 = http://$name.$domain_suffix/$name.crt
crl_url                 = http://$name.$domain_suffix/$name.crl
ocsp_url                = http://ocsp.$name.$domain_suffix:9080
default_ca              = ca_default
name_opt                = utf8,esc_ctrl,multiline,lname,align

[ca_dn]
countryName             = "MX"
organizationName        = "LeanoCompany"
commonName              = "Root CA"
```

```
[ca_default]
home                    = .
database                = $home/db/index
serial                  = $home/db/serial
crlnumber               = $home/db/crlnumber
certificate             = $home/$name.crt
private_key             = $home/private/$name.key
RANDFILE                = $home/private/random
new_certs_dir           = $home/certs
unique_subject          = no
copy_extensions         = none
default_days            = 3650
default_crl_days        = 365
default_md              = sha256
policy                  = policy_c_o_match

[policy_c_o_match]
countryName             = match
stateOrProvinceName     = optional
organizationName        = match
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional
```

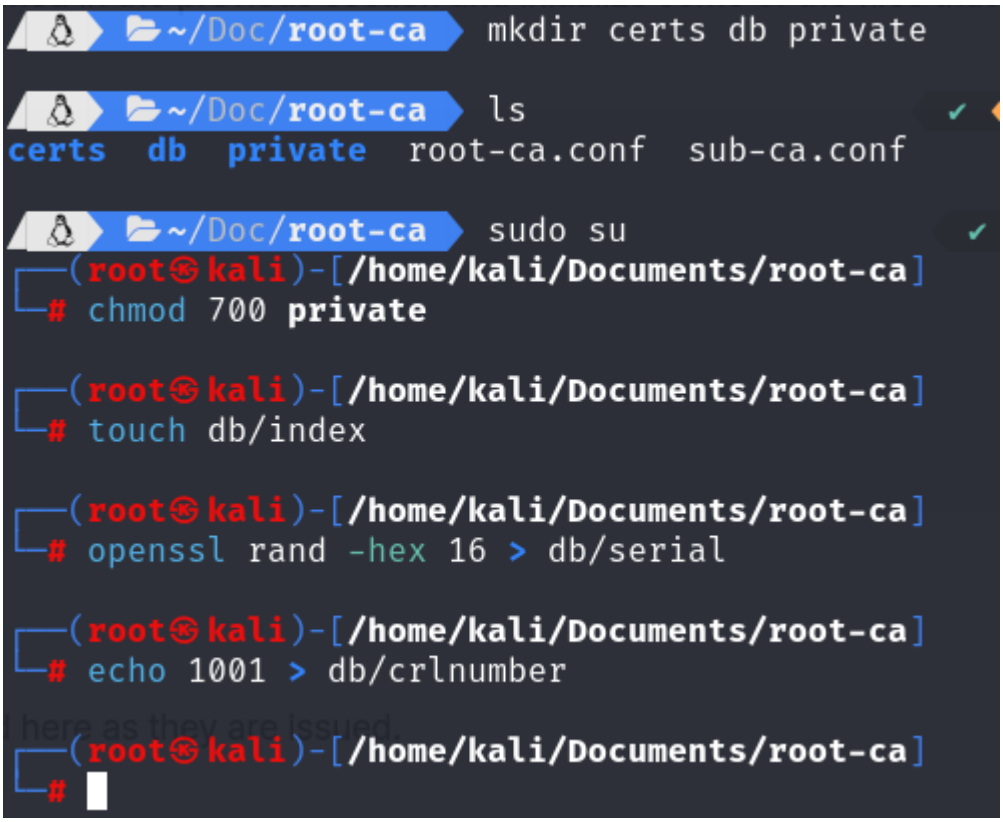
```
[req]
default_bits            = 4096
encrypt_key             = no
default_md              = sha256
utf8                    = yes
string_mask             = utf8only
prompt                  = no
distinguished_name      = ca_dn
```

```
req_extensions          = ca_ext

[ca_ext]
basicConstraints        = critical,CA:true
keyUsage                = critical,keyCertSign,cRLSign
subjectKeyIdentifier    = hash
```

```
[sub_ca_ext]
authorityInfoAccess     = @issuer_info
authorityKeyIdentifier  = keyid:always
basicConstraints        = critical,CA:true,pathlen:0
crlDistributionPoints   = @crl_info
extendedKeyUsage        = clientAuth,serverAuth
keyUsage                = critical,keyCertSign,cRLSign
nameConstraints         = @name_constraints
subjectKeyIdentifier    = hash

[crl_info]
URI.0                   = $crl_url

[issuer_info]
caIssuers;URI.0         = $aia_url
OCSP;URI.0              = $ocsp_url

[name_constraints]
permitted;DNS.0 = leanocompany.com
excluded;IP.0 = 0.0.0.0/0.0.0.0
excluded;IP.1 = 0:0:0:0:0:0:0:0/0:0:0:0:0:0:0:0
```

Creamos las siguientes carpetas le damos permisos para que solo el usuario que lo creo pueda acceder, y ponemos los números de serie con números random así te evitas de problemas.

```
$ mkdir root-ca
$ cd root-ca
$ mkdir certs db private
$ chmod 700 private
$ touch db/index
$ openssl rand -hex 16  > db/serial
$ echo 1001 > db/crlnumber
```



`certs/`

Almacenamiento de certificados; los nuevos certificados se colocarán aquí a medida que se emitan

`db/`

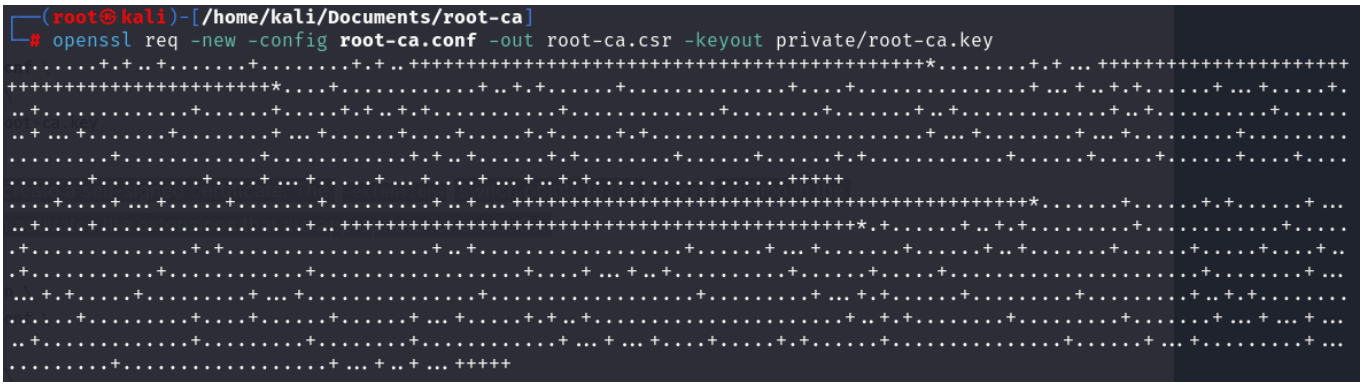
Este directorio se utiliza para la base de datos de certificados (índice) y los archivos que contienen los siguientes números de serie de certificados y CRL. OpenSSL creará algunos archivos adicionales según sea necesario.

`private/`

Este directorio almacenará las claves privadas, una para la CA y la otra para el respondedor de OCSP. Es importante que ningún otro usuario tenga acceso a él. (De hecho, si va a tomarse en serio la CA, la máquina en la que se almacena el material raíz debe tener solo una cantidad mínima de cuentas de usuario).

Para crear la CA raíz, realizamos dos pasos. Primero, generamos la clave y la CSR. Toda la información necesaria se obtendrá del archivo de configuración cuando usemos el modificador -config:

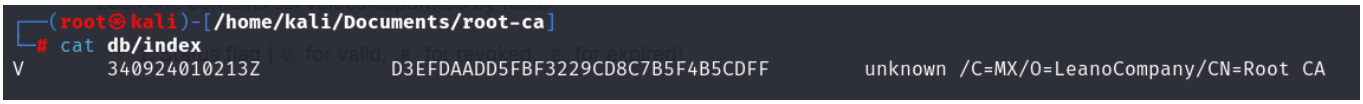
```
openssl req -new -config root-ca.conf -out root-ca.csr -keyout private/root-ca.key
```



En el segundo paso, creamos un certificado autofirmado. El modificador -extensions apunta a la sección ca_ext del archivo de configuración, que activa las extensiones adecuadas para una CA raíz:

```
openssl ca -selfsign -config root-ca.conf -in root-ca.csr -out root-ca.crt
-extensions ca_ext
```

La base de datos en `db/index` es un archivo de texto sin formato que contiene información del certificado, un certificado por línea. Inmediatamente después de la creación de la CA raíz, debe contener solo una línea:



Creating a Subordinate CA

Para sub-ca creamos otra carpeta llamada Sub-ca en esta vamos a repetir los pasos anteriores.

```
[default]
name                    = sub-ca
ocsp_url                = http://ocsp.$name.$domain_suffix:9081

[ca_dn]
countryName             = "MX"
organizationName        = "LeanoCompany"
commonName              = "Sub CA"

[ca_default]
default_days            = 365
default_crl_days        = 30
copy_extensions         = copy
```

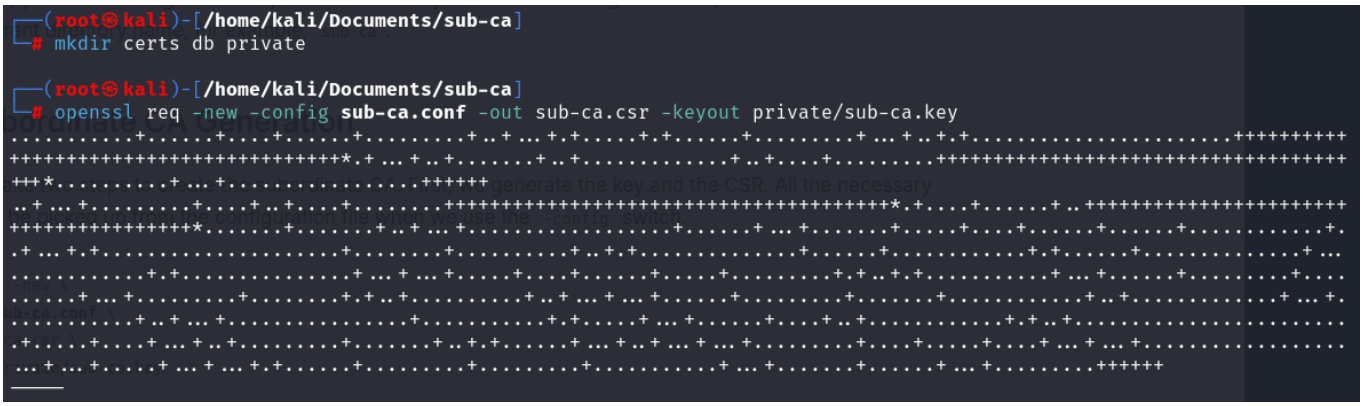
```
[server_ext]
authorityInfoAccess     = @issuer_info
authorityKeyIdentifier  = keyid:always
basicConstraints        = critical,CA:false
crlDistributionPoints   = @crl_info
extendedKeyUsage        = clientAuth,serverAuth
keyUsage                = critical,digitalSignature,keyEncipherment
subjectKeyIdentifier    = hash

[client_ext]
authorityInfoAccess     = @issuer_info
authorityKeyIdentifier  = keyid:always
basicConstraints        = critical,CA:false
crlDistributionPoints   = @crl_info
extendedKeyUsage        = clientAuth
keyUsage                = critical,digitalSignature
subjectKeyIdentifier    = hash
```

Hacemos los mismos pasos pero con sub-ca:

```
$ mkdir sub-ca
$ cd sub-ca
$ mkdir certs db private
$ chmod 700 private
$ touch db/index
$ openssl rand -hex 16  > db/serial
$ echo 1001 > db/crlnumber
```

Como antes, seguimos dos pasos para crear la CA subordinada. Primero, generamos la clave y la CSR. Toda la información necesaria se obtendrá del archivo configuracion.

```
openssl req -new -config sub-ca.conf -out sub-ca.csr -keyout private/sub-
ca.key
```



Mandamos una copia del archivo sub-ca.csr a root-ca

En el segundo paso, conseguimos que la CA raíz emita un certificado. El modificador -extensions apunta a la sección sub_ca_ext en el archivo de configuración, que activa las extensiones adecuadas para la CA subordinada.

```
openssl ca -config root-ca.conf -in sub-ca.csr -out sub-ca.crt -extensions
sub_ca_ext
```

Aquí es donde vamos a hacer la llave para crear el certificado del server, creamos una carpeta llamada key-ca y aquí adentro haremos todo:

```
openssl genpkey -out fd.key -algorithm EC -pkeyopt ec_paramgen_curve:P-256
```

```
┌──(root㉿kali)-[/home/kali/Documents]
└─# mkdir key-ca

┌──(root㉿kali)-[/home/kali/Documents]
└─# cd key-ca

┌──(root㉿kali)-[/home/kali/Documents/key-ca]
└─# openssl genkey -out fd.key -algorithm EC -pkeyout ec_paramgen_curve:P-256
Invalid command 'genkey'; type "help" for a list.

┌──(root㉿kali)-[/home/kali/Documents/key-ca]
└─# openssl genpkey -out fd.key -algorithm EC -pkeyout ec_paramgen_curve:P-256
genpkey: Extra (unknown) options: "pkeyout" "ec_paramgen_curve:P-256"
genpkey: Use -help for summary.

┌──(root㉿kali)-[/home/kali/Documents/key-ca]
└─# openssl genpkey -out fd.key -algorithm EC -pkeyopt ec_paramgen_curve:P-256
```

```
┌──(root㉿kali)-[/home/kali/Documents/key-ca]
└─# openssl genpkey -out fd.key -algorithm EC -pkeyopt ec_paramgen_curve:P-256

┌──(root㉿kali)-[/home/kali/Documents/key-ca]
└─# cat fd.key
-----BEGIN PRIVATE KEY-----
MIGHAgEAMBMGByqGSM49AgEGCCqGSM49AwEHBG0wawIBAQQg77XRvqPB8b9Ztxzx
WwF5MmCVR0S2zusE4vqvmjUqPuahRANCAAT0vdvO4A7OtMACiriH3++q9NH1P4Gz
cvgyMsQt587qf96MIA07PcfwsVJuRsdqiW7AuWO1dtNUcWkVR6HVQtVZ
-----END PRIVATE KEY-----
```

Ya teniendo la llave creamos nuestro firma de certificado que es el archivo que se le tiene que mandar a la autoridad certificadora para que te de tu certificado ya que ahí viene tu información.

```
openssl req -new -key fd.key -out fd.csr
```

Aquí llenas todo lo que te pide con tus datos.

```
┌──(root㉿kali)-[/home/kali/Documents/key-ca]
└─# openssl req -new -key fd.key -out fd.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:MX
State or Province Name (full name) [Some-State]:Jalisco
Locality Name (eg, city) []:.
Organization Name (eg, company) [Internet Widgits Pty Ltd]:LeanoCompany
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:leanocompany.com
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:

┌──(root㉿kali)-[/home/kali/Documents/key-ca]
└─#
```

```
┌──(root㉿kali)-[/home/kali/Documents/key-ca]
└─# cat fd.csr
-----BEGIN CERTIFICATE REQUEST-----
MIIBDTCBswIBADBRMQswCQYDVQQGEwJNWDEQMA4GA1UECAwHSmFsaXNjbzEVMBMG
A1UECgwMTGVhbm9Db21wYW55MRkwFwYDVQQDDBBsZWFub2NvbXBhbnkuY29tMFkw
EwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE9L3bzuAOzrTAAoq4h9/vqvTR9T+Bs3L4
MjLELefO6n/ejCANOz3H8LFSbkbHaoluwLljtXbTVHFpFUeh1ULVWaAAMAoGCCqG
SM49BAMCA0kAMEYCIQDYDBQw6AI125hIIrI5gClDm5F7zCYUzgdGKEM6khQ4vAIh
APs7BxFrfWETrn4qYXySRIZ7dN6UKOIqOk17P3sWAs0D
-----END CERTIFICATE REQUEST-----
```

Para emitir un certificado de servidor, procese una CSR mientras especifica server_ext en el modificador -extensions:

Ya que tenemos nuestro CSR se la mandamos a la autoridad certificadora que seria nuestra sub-ca para crear nuestro certificado.

```
openssl ca -config sub-ca.conf -in fd.csr -out server.crt -extensions
server_ext
```

Y listo ya tendríamos nuestro tres archivos CRT que son :

`root-ca.crt`

`sub-ca.crt`

`server.crt`

Referencias

https://www.feistyduck.com/library/openssl-cookbook/online/openssl-command-line/private-ca.html

https://github.com/ivanr/bulletproof-tls/tree/master/private-ca